

# Scalable Nonlinear Embeddings for Semantic Category-based Image Retrieval

Gaurav Sharma and Bernt Schiele

Max Planck Institute for Informatics, Germany

lastname@mpi-inf.mpg.de

## Abstract

We propose a novel algorithm for the task of supervised discriminative distance learning by nonlinearly embedding vectors into a low dimensional Euclidean space. We work in the challenging setting where supervision is with constraints on similar and dissimilar pairs while training. The proposed method is derived by an approximate kernelization of a linear Mahalanobis-like distance metric learning algorithm and can also be seen as a kernel neural network. The number of model parameters and test time evaluation complexity of the proposed method are  $O(dD)$  where  $D$  is the dimensionality of the input features and  $d$  is the dimension of the projection space—this is in contrast to the usual kernelization methods as, unlike them, the complexity does not scale linearly with the number of training examples. We propose a stochastic gradient based learning algorithm which makes the method scalable (w.r.t. the number of training examples), while being nonlinear. We train the method with up to half a million training pairs of 4096 dimensional CNN features. We give empirical comparisons with relevant baselines on seven challenging datasets for the task of low dimensional semantic category based image retrieval.

## 1. Introduction

Learning distance metrics for comparing multi-dimensional vectors is a fundamental problem. If a perfect task adaptive distance metric is available, many important computer vision problems such as image (object, scene etc.) classification and retrieval become trivial using nearest neighbor search [12, 51]. Metric learning is also applicable to many other important computer vision tasks requiring vector comparisons [2, 14, 23, 25, 27, 26].

In the present paper, we are interested in learning a distance metric by embedding the vectors into a low dimensional Euclidean space. We work with pairwise constraints of the form  $\{(\mathbf{x}_i, \mathbf{x}_j, y_{ij})\}$  where  $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^D$  are the feature vectors and  $y_{ij} = +1$  if they are semantically similar and should have a small distance, and  $y_{ij} = -1$  otherwise. This is a practical setting as obtaining such side information

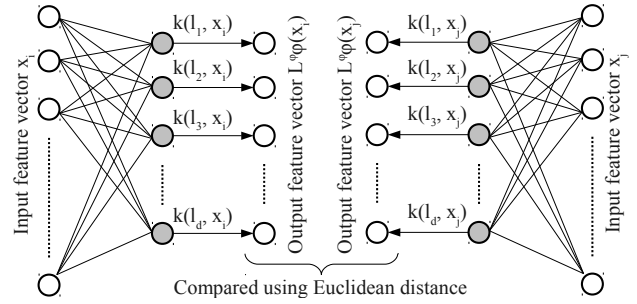


Figure 1. The proposed method seen as a kernel neural network with one hidden layer (shaded). The input features, for the test pair, are passed through the network respectively and then compared using Euclidean distance.

is easier, e.g. by getting user feedback, than annotating all the vectors for their classes. Moreover, in such scenarios, as no class specific models are learned, the distances and embeddings learned are generic.

Metric learning has attracted substantial attention in the machine learning community [8, 11, 12, 51] and has specifically achieved much success in computer vision e.g. for image auto-annotation [13], face verification [27], visual tracking [23], person reidentification [2] and nearest neighbor based image classification [26]. While initial work on metric learning was mostly for learning a linear Mahalanobis-like distance, nonlinear metric learning has also been explored [8, 15, 27, 34]. Linear methods have been extended to be nonlinear e.g. by *kernelization* [8, 15, 27] (we discuss in detail in §2) and other nonlinear learning methods have been proposed [6, 18, 29]. However, such nonlinear methods have not been demonstrated to be scalable (to order of millions of training points in spaces of order of thousands of dimensions). Similar to kernel support vector machines (SVM), the complexity (number of model parameters and evaluation time) of kernelized versions is often linear in the number of training examples<sup>1</sup>.

<sup>1</sup>More precisely, for kernel SVMs, the complexity is linear in the  $N_{sv}$  (number of support vectors) whose expectation is bounded below by  $(\ell - 1)E(p)$  where  $E(p)$  is the expectation of the probability of error on a test vector and  $\ell$  is the number of training examples [4, 45]. Hence, the complexity can be expected to scale approximately linearly with the number of training examples.

In view of such undesirable scaling, we make the following contributions in this paper. (i) Inspired by recent work on efficient nonlinear SVMs [38, 39], we propose a novel metric learning method using kernels. The proposed method can also be seen as a kernel neural network with one hidden layer (Fig. 1), trained to optimize the verification objective i.e. bringing similar pairs close and pushing dissimilar ones far. We propose to use an efficient stochastic gradient descent (SGD) algorithm for training the system. Use of SGD combined with the fact that the complexity (number of model parameters and evaluation time) of the proposed method does not depend on the number of training examples makes the method scalable w.r.t. the number of training examples. While in the present paper we work with nonlinearity based on the popular  $\chi^2$  kernel [47], the method is generalizable to kernels whose derivatives can be computed analytically. (ii) We show consistent improvement obtained by the method for the task of semantic category based retrieval with seven challenging publicly available image datasets of materials, birds, human attributes, scenes, flowers, objects and butterflies. Our experimental results support the datasets. (iii) We also demonstrate scalability by training with order of millions of training pairs of 4096 dimensional state-of-the-art CNN features [19, 46] and compare with five existing competitive baselines.

## 2. Related work and background

Metric learning has been an active topic of research (we encourage the interested reader to see [3, 20] for extensive surveys) with applications to face verification [16], person reidentification [2], image auto-annotation [13], visual tracking [23], nearest neighbor based image classification [26] etc. in computer vision. Starting from the seminal paper of Xing et al. [52], many different approaches for learning metrics have been proposed e.g. [8, 11, 12, 15, 27, 34, 35, 43, 44, 50].

Different types of supervision has been used for learning metrics. While some methods require class level supervision [32], others only require triplet constraints, i.e.  $\{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)\}$ , where  $\mathbf{x}_i$  should be closer to  $\mathbf{x}_j$  than to  $\mathbf{x}_k$  [50], and others still, only pairwise constraints, i.e.  $\{(\mathbf{x}_i, \mathbf{x}_j, y_{ij})\}$  where  $y_{ij} = +1$  if  $(\mathbf{x}_i, \mathbf{x}_j)$  are similar and  $y_{ij} = -1$  if they are dissimilar [27].

Most of the initial metric learning methods were linear, e.g. the semidefinite programming formulation by Xing et al. [52], large margin formulation for  $k$ -NN classification by Weinberger et al. [50], ‘collapsing classes’ formulation (make the distance between vectors of same class zero and between those of different classes large) of Globerson and Roweis [11] and neighbourhood component analysis of Goldberger et al. [12].

Towards scalability of metric learning methods, Jain et al. [17] proposed online metric learning and more recently

Simonyan et al. [41] proposed to use stochastic gradient descent for face verification problem.

Works also reported learning nonlinear metrics. Many linear metric learning approaches were shown to be kernelizable [8, 15, 27, 34, 35, 43, 44]. Tsang and Kwok [44] proposed a metric learning problem similar to the  $\nu$ -SVM [33], which they solved in the dual allowing the use of kernels. Schultz and Joachims [34] proposed support vector machine based algorithm which was accordingly kernelized while Chatpatanasiri et al. [5] proposed to use kernel PCA for nonlinear metric learning. Mignon and Jurie [27] proposed to use kernels for metric learning with sparse pairwise constraints with a logistic loss based objective function—we are interested in a similar weakly supervised setting and we discuss this in more detail in the next section (§2.1).

Other than nonlinearity via kernelization, alternate nonlinear forms of metrics have also been studied e.g. based on variations and adaptations of neural networks [6, 32], boosting [18] binary codes with hamming distances [29]. Weinberger et al. [51] also proposed to use local metric learning for introducing nonlinearity in the learnt metric.

### 2.1. Background: Supervised discriminative metric learning with pairwise constraints

Given a dataset  $\mathcal{X}$  of positive and negative pairs of vectors i.e.  $\mathcal{X} = \{(\mathbf{x}_i, \mathbf{x}_j, y_{ij}) | (i, j) \in \mathcal{I} \subset \mathbb{N}^2\}$ , with  $\mathbf{x}_i \in \mathbb{R}^D$  and  $y_{ij} \in \{-1, +1\}$  and  $\mathcal{I}$  being an index set, the task is to learn a distance function in  $\mathbb{R}^D$ .

Many metric learning approaches learn, from  $\mathcal{X}$ , a Mahalanobis-like metric parametrized by matrix  $M \in \mathbb{R}^{D \times D}$ , i.e.

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M (\mathbf{x}_i - \mathbf{x}_j). \quad (1)$$

$M$  is required to be symmetric positive semi-definite (PSD) matrix, for the distance to be a valid metric, and hence can be factorized as  $M = \tilde{L}^\top \tilde{L}$ , with  $\tilde{L} \in \mathbb{R}^{d \times D}$  and  $d \leq D$ . The metric learning can then be seen as learning a projection upon which the comparison is done using Euclidean distance in the resulting space i.e.

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = \|\tilde{L}\mathbf{x}_i - \tilde{L}\mathbf{x}_j\|_2^2. \quad (2)$$

Learning such distance function has been achieved by using, among other methods, optimization of probabilistic objectives (based on likelihood) or loss functions based on the max-margin principle e.g. for the task of face verification in computer vision [13, 27, 41]. Here, we minimize the objective with hinge loss, i.e.

$$\min_{\tilde{L}, b} F = \sum_{\mathcal{X}} \max(0, 1 - y_{ij}\{b - d^2(\mathbf{x}_i, \mathbf{x}_j)\}), \quad (3)$$

which aims to learn  $\tilde{L}$  such that the positive pairs are at distances less than  $b - 1$ , to each other, while the negatives are

at distances greater than  $b + 1$ , with  $b$  being the bias parameter i.e. a threshold on the distance between two vectors to decide if they are same or not. Explicit regularization is absent as often  $d \ll D$  i.e. the rank  $d$  of the learnt metric  $M = \tilde{L}^\top \tilde{L}$  is fixed to be small.

While the distance function learned as above is linear, the problem may be complex and require nonlinear distance function. A popular way of learning a nonlinear distance function is by kernelizing the metric, as inspired by the traditional kernel based methods e.g. KPCA and KLDA; invoke representer theorem like condition and write the rows of  $\tilde{L}$  as linear combinations of the input vectors i.e.  $\tilde{L} = AX^T$  (where  $X$  is the matrix of all vectors  $\mathbf{x}_i$  as columns). Noticing that the distance function in Eq. 2 depends only on the dot products of the vectors, allows non-linearizing the algorithm as follows. Mapping the vectors with a non-linear *feature map*  $\phi : \mathbb{R}^D \rightarrow \mathcal{F}$  and then using the *kernel trick* i.e.  $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{F}}$ , we can proceed as follows,

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = \|AX_\phi^T \phi(\mathbf{x}_i) - AX_\phi^T \phi(\mathbf{x}_j)\|^2 = \|A(\mathbf{k}_i - \mathbf{k}_j)\|^2, \quad (4)$$

where,  $X_\phi = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \dots, \phi(\mathbf{x}_n)]$  is the matrix of  $\phi$  mapped vectors and

$$\mathbf{k}_t = X_\phi^T \mathbf{x}_t = [k(\mathbf{x}_1, \mathbf{x}_t), k(\mathbf{x}_2, \mathbf{x}_t), \dots, k(\mathbf{x}_n, \mathbf{x}_t)]^T \quad (5)$$

is the  $t^{th}$  column of the kernel matrix. Such reasoning was used by Mignon and Jurie [27] recently. While this is a successful way of non-linearizing the algorithm, it is costly and not scalable as training requires the whole kernel matrix.

### 3. Proposed method

We now give the details of the proposed nonlinear embeddings by approximate kernelization of Mahalanobis-like distance metric learning.

Continuing from the discussion in the previous section 2.1, we note that the rows of  $\tilde{L}$  can be thought of as a basis set (albeit not necessarily orthogonal) on which the test vectors are projected by taking dot products. After projection the comparison is simply done using the Euclidean distance. Now consider again a nonlinear feature map  $\phi : \mathbb{R}^D \rightarrow \mathcal{F}$ , and the corresponding kernel function  $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{F}}$ ; we can view the projection, with matrix  $L^\phi$  in the  $\phi$  mapped space  $\mathcal{F}$ , as

$$L^\phi \phi(\mathbf{x}) = [\langle \ell_1^\phi, \phi(\mathbf{x}) \rangle_{\mathcal{F}}, \dots, \langle \ell_d^\phi, \phi(\mathbf{x}) \rangle_{\mathcal{F}}], \quad (6)$$

where  $\ell_t^\phi \in \mathcal{F}$  are the rows of  $L^\phi$ . Instead of writing each  $\ell^\phi$  as linear combinations of  $\{\phi(\mathbf{x}_i)\}$ , as done traditionally, we take an alternate route and make an approximating assumption as follows. We recall the concept of pre-image, in  $\mathbb{R}^D$ , of a vector in  $\mathcal{F}$  i.e.  $\mathbf{x} \in \mathbb{R}^D$  corresponding to a feature

---

#### Algorithm 1 SGD for proposed nonlinear metric learning

---

```

1: Given: Training set ( $\mathcal{X}$ ), margin ( $m$ ), learning rate ( $r$ )
2: Initialize:  $b = 1$ ,  $L \leftarrow \text{random}(-0.5, 0.5)$ 
3: for all  $i = 1, \dots, \text{niters}$  do
4:   Randomly sample a training pair  $(\mathbf{x}_i, \mathbf{x}_j, y_{ij}) \in \mathcal{X}$ 
5:   Compute  $d_\phi^2(\mathbf{x}_i, \mathbf{x}_j)$  using Eq. 8
6:   if  $y_{ij}(b - d_\phi^2(\mathbf{x}_i, \mathbf{x}_j)) < m$  then
7:     for all  $t = 1, \dots, d$  do
8:       // ref. Eq. 10
9:        $\ell_t \leftarrow \ell_t - ry_{ij}(k_i^t - k_j^t)(\nabla_{\ell_t} k_i^t - \nabla_{\ell_t} k_j^t)$ 
10:    end for
11:  end if
12: end for

```

---

space vector  $\mathbf{x}^\phi \in \mathcal{F}$  such that  $\phi(\mathbf{x}) = \mathbf{x}^\phi$ , which has been studied in the past in the context of different kernel methods [4, 21]. We assume that there exists  $\ell_t \in \mathbb{R}^D$  such that either  $\phi(\ell_t) = \ell_t^\phi$  i.e.  $\ell_t \in \mathbb{R}^D$  is the pre-image of  $\ell_t^\phi \in \mathcal{F}$  or, if such pre-image doesn't exist, then  $\phi(\ell_t) \approx \ell_t^\phi$  i.e.  $\ell_t \in \mathbb{R}^D$  is an approximation for the pre-image of  $\phi(\ell_t)$ .

Once we have  $\phi(\ell_t) = \ell_t^\phi$ , we can then write

$$L^\phi \phi(\mathbf{x}) = [\langle \phi(\ell_1), \phi(\mathbf{x}) \rangle_{\mathcal{F}}, \dots, \langle \phi(\ell_d), \phi(\mathbf{x}) \rangle_{\mathcal{F}}] \\ = [k(\ell_1, \mathbf{x}), k(\ell_2, \mathbf{x}), \dots, k(\ell_d, \mathbf{x})]. \quad (7)$$

Intuitively, what we did here is that instead of a linear projection of the test vector on the rows of  $\tilde{L}$  as in the linear case (§2.1 above), we now do a ‘nonlinear projection’ on the rows of  $L = [\ell_1^T, \ell_2^T, \dots, \ell_d^T]^T$ . This can also be seen in the similar spirit as dimensionality reduction using nonlinear kernel methods e.g. in kernel PCA the principal components come out to be linear combinations of  $\phi$  mapped input vectors i.e.  $\sum_i \gamma_i \phi(\mathbf{x}_i)$ , and the test vectors are projected nonlinearly to these principal components. Similarly, our method can be seen as a way of doing nonlinear dimensionality reduction with, as we will detail in the following, discriminative supervised learning using similar and dissimilar pairs annotations.

Following our assumption, the distance function computed in the feature space becomes,

$$d_\phi^2(\mathbf{x}_i, \mathbf{x}_j) = \|L^\phi \phi(\mathbf{x}_i) - L^\phi \phi(\mathbf{x}_j)\|^2 \\ = \sum_{t=1}^d (k(\ell_t, \mathbf{x}_i) - k(\ell_t, \mathbf{x}_j))^2. \quad (8)$$

With this formulation, the parameters to be learned are the elements of the matrix  $L$  i.e.  $\ell_t \in \mathbb{R}^D \forall t = 1, \dots, d$ . Note that this matrix is different from the  $\tilde{L}$  matrix in the linear distance function case above in §2.1. We propose to learn  $L$  with a standard max margin hinge loss based objective. The optimization problem thus takes the form,

$$\min_L F = \sum_{\mathcal{X}} \max(0, m - y_{ij}\{b - d_\phi^2(\mathbf{x}_i, \mathbf{x}_j)\}), \quad (9)$$

where the fixed margin of 1 is replaced by a free parameter  $m$ . This is important as depending on the kernel used the distances may be bounded from above e.g. with histogram based kernels commonly used in computer vision, say  $\chi^2$  kernel, the maximum distance between two vectors (histograms) is bounded by zero from below and unity from above. Hence, clearly the bias  $b$  (recall that this is like a threshold on the distances, to decide if a pair is same or not) has to be less than unity and consequently the margin has to be less than unity as well. As we will show later in experiments, the method is not very sensitive to these parameters and we fixed it once for different datasets.

Finally, we substitute for the distance function in feature space using Eq. 8 and propose to optimize the objective efficiently using SGD. At each iteration we sample an annotated training pair and update the  $L$  matrix based on the sub-gradients of the objective function. While the objective function is nonlinear we find that the optima obtained with our SGD implementation perform consistently and well in practice, without per instance/database tuning.

Alg. 1 gives the pseudo code of the procedure used for generating the experimental results in this paper. The sub-gradients required for the SGD iterations in the algorithm are calculable analytically and are as follows,

$$\begin{aligned} \nabla_{\ell_t} F(\{(\mathbf{x}_i, \mathbf{x}_j, y_{ij})\}; L, b) \\ = \begin{cases} 0 & \text{if } y_{ij}\{b - d_\phi^2(\mathbf{x}_i, \mathbf{x}_j)\} \geq m \\ 2y_{ij}(k_i^t - k_j^t)(\nabla_{\ell_t} k_i^t - \nabla_{\ell_t} k_j^t) & \text{otherwise,} \end{cases} \end{aligned} \quad (10)$$

where  $k_i^t = k(\ell_t, \mathbf{x}_i)$  and  $\nabla_b F(\{(\mathbf{x}_i, \mathbf{x}_j, y_{ij})\}; L, b) = 0$  if  $y_{ij}\{b - d_\phi^2(\mathbf{x}_i, \mathbf{x}_j)\} \geq 1$  and  $-y_{ij}$ , otherwise. As a last detail, we use the shifted  $\chi^2$  kernel, shown to be quite effective for image classification etc. [47], given by

$$k(\mathbf{x}, \mathbf{y}) = \sum_{c=1}^D \frac{2x_c y_c}{|x_c| + |y_c|}. \quad (11)$$

The gradient for the  $\chi^2$  kernel is also analytically calculable and is given by,

$$\nabla_{\ell} k(\ell, \mathbf{x}) = \frac{2x_d |x_d|}{(|x_d| + |\ell_d|)^2}. \quad (12)$$

The complexity of the stochastic updates is  $O(dD)^2$  (kernel evaluations with  $\ell_t \forall t = 1, \dots, d$ ), while the complexity of similar updates with the traditional parametrization will be  $O(ND)$  where  $N$  is the number of distinct training vectors, as the  $\ell_t$  are linear combinations of  $\phi(\mathbf{x}_i)$ . Since  $d$  is fixed and small ( $d \ll N$ ) the updates are relatively inexpensive and the algorithm is, thus, scalable to

<sup>2</sup>While there are kernels whose computation complexity is not linear in  $D$  e.g. kernel based on the earth movers distance (EMD) [31], we work here only with those with linear complexity.

order of millions of training points. Also, since the learning is using a SGD based algorithm which takes the training pairs sequentially, it can also be applied in an online setting where the training examples are only available with time.

Another important salient feature of the algorithm is that it can directly work with high dimensional vectors and does not require them to be compressed (preprocessed) first with unsupervised methods, e.g. PCA [51] or KPCA [5]. This potentially allows the method to exploit the full representative power of the high dimensional space of the vector for the current discrimination task, which might be otherwise lost in the case of unsupervised dimensionality reduction as a preprocessing.

## 4. Experimental results

**Datasets.** We report experiments with seven publicly available datasets: Pascal VOC 2007 [10], Scene-15 [22], Flickr Materials (FMD) [36], Oxford 102-Flowers [28], Leeds Butterflies [49], Caltech UCSD Birds 200-2011 (CUB) [48] and Human Attributes (HAT) [37]. Tab. 1 gives the statistics, i.e. number of classes and number of training, validation and testing images, for the seven datasets. We use the provided train+val sets for training and test set for testing where available. Otherwise, for Scene-15 we randomly take 100 images per class for training and rest for testing and for Leeds Butterflies we take the first 20 images of each class for testing and rest for training. We use only the classes with less than 1000 positive images in the HAT dataset, as including the images with high number of positives was giving saturated results, while respecting the train/val/test split provided.

**Image features.** CNN features have been quite successful in image classification after the seminal work of Krizhevsky et al. [19], have been competitive for many different computer vision tasks [30]. Thus, we use CNN features using the MatConvNet [46] library, with the 16 layer model [42] which is pre-trained on the ImageNet dataset [9]. We use the outputs of the last fully connected layer after linear rectification i.e. our features are non-negative. To validate the baseline implementation we used the extracted 4096 dimensional CNN features with linear SVM to perform the classification task for the different datasets. Tab. 1 gives the performance and those of the best method on the dataset. We see that the features used here perform competitively to existing methods. Hence, we conclude that the features we use in the experiments are relevant and comparable to the state-of-the-art.

**Baselines.** We report results with five baselines. In all cases the final vectors are compared using Euclidean distance. As a reference, we take the full ( $\ell_2$  normalized) 4096 dimensional CNN features (denoted ‘No proj’ in the figures) with Euclidean distance. Such a system was recently shown to be



Dataset	#classes	Statistics		Performances	
		#train+val	#test	Present	Existing
Scene-15 <sup>1</sup> [22]	15	1500	2985	90.3	91.6 [53]
Flickr Materials <sup>1</sup> [36]	10	500	500	79.6	82.8 [7]
Leeds Butterflies <sup>1</sup> [49]	10	632	200	99.0	96.4 [24]
Pascal VOC 2007 <sup>2</sup> [10]	20	5011	4952	86.1	89.7 [42]
Human Attributes <sup>2</sup> [37]	27	7000	2344	55.4	59.7 [40]
Oxford 102-Flowers <sup>1</sup> [28]	102	2020	6149	84.3	86.8 [30]
Caltech UCSD Birds <sup>1</sup> [48]	200	5994	5794	63.1	69.1 [7]

Table 1. Statistics of the datasets used for the experiments, along with the performances (<sup>1</sup> mean class accuracy or <sup>2</sup> mean average precision) of the features used (with SVM) vs. existing methods. The performances serve to demonstrate that the features we use are competitive.

competitive for instance retrieval [1]. As the first baseline, we do Principal Component Analysis (PCA) based dimensionality reduction. As the second baseline, we learn a metric using the Neighborhood Component Analysis (NCA) [12]. As the third and fourth baseline, we learn a metric using the Large Margin Nearest Neighbor (LMNN) [51] algorithm with (i) vectors reduced to dimension  $d$  with PCA, and learning a square projection metric, denoted LMNN(s), and (ii) vectors reduced to 256 dimensions (98% variance on average) with PCA (for efficiency) and then learning a rectangular projection matrix, denoted LMNN(r). HAT and VOC 2007 datasets have some images which have multiple labels; for training NCA and LMNN baselines such images were removed from the training set as these algorithm use a multi-class supervision. Publicly available code<sup>3</sup> is used for NCA and LMNN algorithms. As the final baseline we use the linear metric learning (ML) algorithm as described in §2.1 with similar objective function (modulo nonlinearity) and the same training data as the proposed method.

**Evaluation.** We report results for the retrieval setting. We use the train+val sets of the datasets for training our baselines and the proposed nonlinear method. We use each test image as a query and the rest of the test images as the gallery and report the mean precision@ $K$  (mprec@ $K$ ) i.e. average of precision@ $K$  is computed for queries of a given class, averaged over the classes. The CNN features for the test images are transformed by the respective methods and are compared with Euclidean distance.

**Implementation details.** We fix the number of iterations to one million. We sample (up to) 500,000 pairs of similar and dissimilar vectors from the train+val set for training both the baseline linear metric learning and the proposed nonlinear metric learning (NML). The baseline methods LMNN and NCA do not use similar pairwise constraints but constraints derived using class labels. The vectors were  $\ell_2$  normalized for all baselines<sup>4</sup> and were  $\ell_1$  normalized

<sup>3</sup><http://www.cse.wustl.edu/~kilian/code/lmnn/lmnn.html>

<sup>4</sup>We also tried  $\ell_1$  normalization with  $\ell_1$  and  $\chi^2$  distances for the reference 4096 dimensional vectors, the results were similar (see supplementary material).

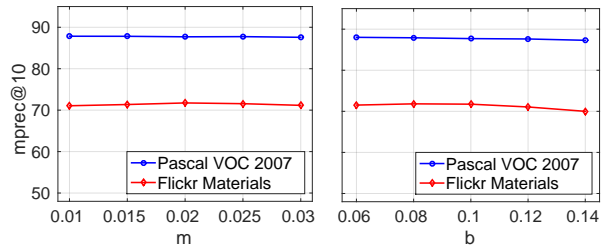


Figure 2. Performance with varying margin  $m$  (fixed  $b = 0.1$ ) and bias  $b$  (fixed  $m = 0.02$ ), for the proposed method ( $d = 64$ ).

for the proposed method (as the  $\chi^2$  kernel is based on histograms). The bias and margin for the linear metric learning baseline were fixed to  $b = 1$  and  $m = 0.2$  while those for the proposed nonlinear method were fixed to  $b = 0.1$  and  $m = 0.02$ . These values were chosen based on preliminary experiments on the VOC 2007 validation set and were kept *constant for all experiments* reported i.e. no dataset specific tuning was done. The algorithm is not very sensitive to the choice of  $m$  and  $b$ , Fig. 2 show the performances for a range of  $m$  and  $b$  values (with the other fixed resp.) on the Flickr Materials [36] and Pascal VOC 2007 [10] datasets. The bias and margin need to be kept sufficiently low. The reason for low values of bias, and resp. margin, for the nonlinear method is that since the vectors are  $\ell_1$  normalized histograms, the maximum score they can give with the  $\chi^2$  kernel is 1 (which only happens with the vector itself) and hence the scale of the distances is expected to be less than 1 for the proposed nonlinear method.

#### 4.1. Quantitative results

We present some results for typical setting and analyse them. We refer the reader to the supplementary material for additional results.

Fig. 3 shows the performance of the different methods for different projection dimensionality  $d \in \{8, 16, 32, 64\}$ , with a fixed  $K = 10$ , i.e. the number of top images retrieved. Fig. 4 shows the performance for different  $K \in [1, 50]$  with a fixed  $d = 16$ . The Birds and Butterflies datasets have  $n^+ = 30$  and  $n^+ = 20$  images per class and hence we only report for  $K$  up to 29 and 19 for them, respectively.

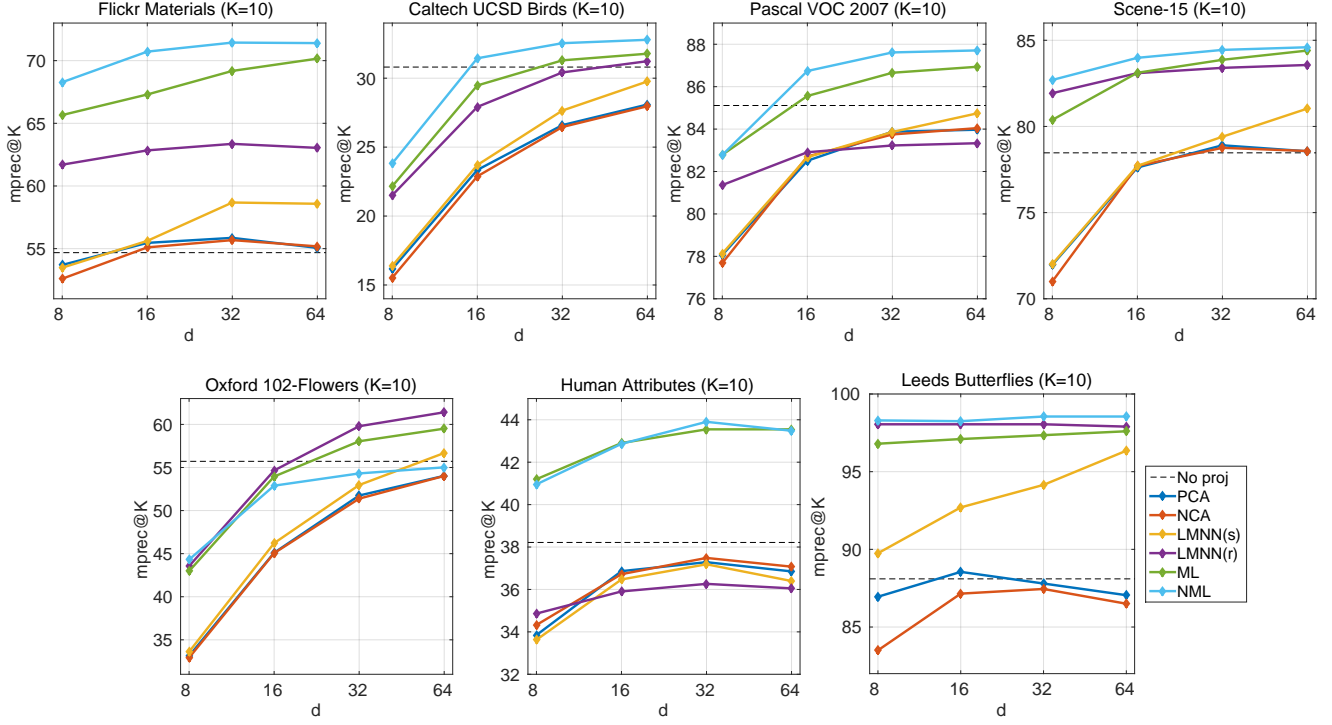


Figure 3. Comparisons of methods on the different datasets for number of top retrievals  $K = 10$ , and projection dimension  $d \in \{8, 16, 32, 64\}$  (see §4.1).

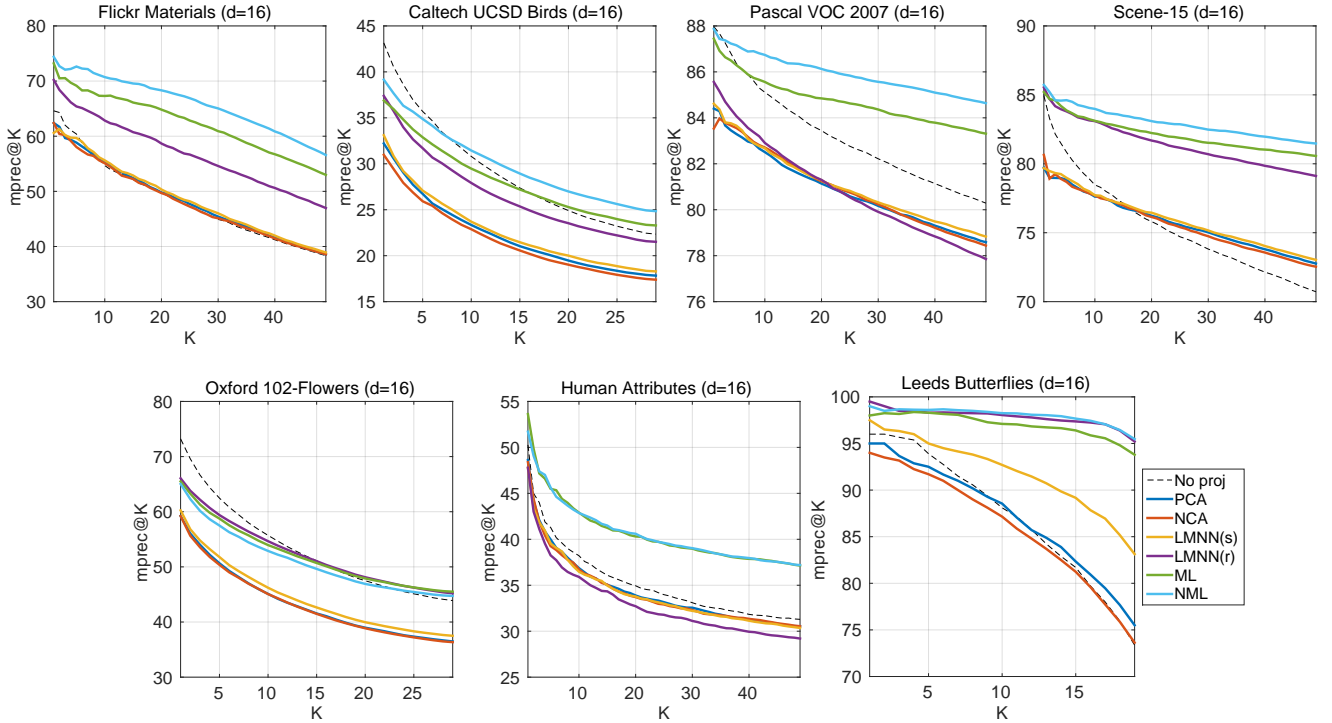


Figure 4. Comparisons of methods on the different datasets for number of top retrievals  $K \in [1, \max(50, n^+ - 1)]$  and projection dimension  $d = 16$  (see §4.1).

We observe, from Fig. 3, for  $K = 10$  i.e. precision for the top 10 retrieved images, that among the baselines, supervised LMNN is generally better than supervised NCA and unsupervised PCA while the ML baseline is the most competitive. The proposed NML is generally better or at least as good as the baselines, notably the most competitive ML baseline. It is outperformed by the baselines only on the Flowers dataset at relatively higher dimensional (e.g.  $d = 32, 64$ ) projections. The improvements are consistent over Materials, Birds, Objects and Scenes datasets for all  $d \in \{8, 16, 32, 64\}$ . For the Human Attributes, NML does not improve the baseline ML, but is essentially similar to it i.e. does not deteriorate either. On the Flowers dataset NML is outperformed by the baselines at higher dimensions. Further, from Fig. 4 we note a similar general trend, for a fixed projection dimension  $d = 16$  while observing the precision at varying number of  $K$  top retrievals. The proposed NML consistently performs better on the Materials, Birds, Objects and Scenes datasets while on the Human Attributes and Butterflies datasets it performs similar, compared to the baselines.

In addition to the results shown here, in general NML was found to be better or at least similar to the baseline ML methods (see supplementary material). The few cases where it is not better are those at high projection dimensions  $d$ , which could be explained by the higher need of nonlinearity of the model at lower dimensions, where an equivalent linear model with similar number of parameters may not suffice, while at higher dimensions, the larger number of parameters for the linear method are sufficient for the task. Thus, we conclude that except in a single case, i.e. the Flowers dataset at relatively higher dimensional projections, NML improves over the reported baselines demonstrating the benefit of the proposed scalable nonlinear embedding and efficient SGD based learning thereof.

Tab. 2 gives typical performances of the kernelized version (§2.1) of baseline ML (kML) vs. the proposed NML, on the Materials dataset. NML obtains similar performances while being  $O(d)$  at test time cf.  $O(N)$  for kML.

It is also interesting to note the performance of the method after compression compared to using the full 4096 dimensional CNN feature without any compression. In both Fig. 3 and Fig. 4 the dotted black line shows this performance as a reference. We see that more often than not, the discriminatively learnt NML projection improves the performance over the full features. For the Materials, Human Attributes, Scenes, and Butterflies datasets this improvement is significant. In the case of Birds and VOC 2007 Objects the performance drops below the reference for low dimension projections while gradually improving at higher dimensional projections. Only in the case of the Flowers dataset, NML projection does not improve this reference but is roughly equal to it at higher dimensions, i.e.  $d = 64$ .

d	kML	NML
8	69.6	68.3
16	70.8	70.7
32	70.9	71.5
64	71.4	71.4

Table 2. Performance (mprec@10) of the kernelized metric learning vs. proposed nonlinear metric learning, on the Flickr Materials dataset, for different projection dimensions  $d$ .

Thus, we conclude that the proposed NML is also beneficial to improve performance of the reference system with full feature vector (4096 dimensional) without compression.

## 4.2. Qualitative results

We show some example retrievals in Fig. 5 using (i) the baseline linear metric learning based projections and (ii) the proposed nonlinear embeddings. For each pair of rows, the first one corresponds to method (i) and second one to (ii). The first image is the query and the rest of the images are retrievals by the respective methods, sorted by their scores. The vectors after projections for both the methods are 8 dimensional floating point numbers i.e. it takes 32 bytes to store one image. The goal is to retrieve images of the same semantic category as the example query e.g. in the first example, images of ‘fabrics’ are retrieved. Similarly other retrievals are for specific types of birds, scenes, butterflies etc. Note the variations in the appearance of the retrieval results, which can be attributed to the combination of the strong CNN based appearance features and discriminatively learned embeddings.

## 4.3. Computation times

The training time (with 500,000 training pairs) is about an hour, on a single core of 3 Ghz CPU running linux, for proposed method while the linear ML with exactly the same training data takes around five minutes ( $d = 8$  for both). For testing (on 500 test images), ML takes 52 ms while the proposed method takes 258 ms, in a similar setup. These times exclude the feature extraction time. Both implementations are in MATLAB and not optimized for performance, however, we suspect ML has an advantage as it depends only on matrix operations which use optimized low-level libraries.

## 5. Discussion and conclusion

We presented a novel method for supervised discriminative distance learning. The method performs nonlinear embedding of the high dimensional vectors into a Euclidean space where the vectors are compared using standard  $\ell_2$  distance. The method is derived from an approximate kernelization of Mahalanobis-like distance metric learning. We proposed an efficient and scalable stochastic gradient based learning method and reported results for the task of semantic category based image retrieval on seven publicly available challenging image datasets of materials, birds, human attributes, scenes, flowers, objects and butterflies. Our





Figure 5. Example retrievals for a given query image (leftmost single image), using the baseline linear metric learning vs. the proposed nonlinear embeddings (first and second rows of each pair resp.). Both methods project to just 8 dimensions i.e. 32 bytes per image. The correctly (incorrectly) retrieved images are highlighted using green (red) bounding boxes.

experimental results support the conclusion that the proposed method is consistently better than standard competitive baselines specially in the case of low dimensional projections. We also made the observation that doing such supervised discriminative dimensionality reduction also improves the performance of the reference system which uses the full high dimensional feature without any compression.

The proposed method used a classic kernel ( $\chi^2$  kernel) for the nonlinearity, exploring alternate, even data driven/learned kernels within a similar framework seems an interesting future direction to us.

## References

- [1] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014.
- [2] A. Bedagkar-Gala and S. K. Shah. A survey of approaches and trends in person re-identification. *IVC*, 32(4):270–286, 2014.
- [3] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *arXiv:1306.6709*, 2013.
- [4] C. J. C. Bruges. Simplified support vector decision rules. In *ICML*, 1996.
- [5] R. Chatpatanasiri, T. Korsrilabutr, P. Tangchanachaianan, and B. Kijirikul. A new kernelization framework for mahalnobis distance learning algorithms. *Neurocomputing*, 73(10):1570–1579, 2010.
- [6] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.
- [7] M. Cimpoi, S. Maji, and A. Vedaldi. Deep convolutional filter banks for texture recognition and segmentation. In *CVPR*, 2015.
- [8] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, 2007.
- [9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. <http://www.pascal->



- network.org/challenges/VOC/voc2007/workshop/index.html, 2007.
- [11] A. Globerson and S. Roweis. Metric learning by collapsing classes. In *NIPS*, 2005.
  - [12] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *NIPS*, 2004.
  - [13] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *CVPR*, 2009.
  - [14] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? Metric learning approaches for face identification. In *ICCV*, 2009.
  - [15] S. C. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma. Learning distance metrics with contextual constraints for image retrieval. In *CVPR*, 2006.
  - [16] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
  - [17] P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *NIPS*, 2008.
  - [18] D. Kedem, S. Tyree, K. Q. Weinberger, F. Sha, and G. Lanckriet. Non-linear metric learning. In *NIPS*, 2012.
  - [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
  - [20] B. Kulis. Metric learning: A survey. *FTML*, 5(4):287–364, 2012.
  - [21] J.-Y. Kwok and I. W. Tsang. The pre-image problem in kernel methods. *IEEE Trans. NN*, 15(6):1517–1525, 2004.
  - [22] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.
  - [23] X. Li, C. Shen, Q. Shi, A. Dick, and A. van den Hengel. Non-sparse linear representations for visual tracking with online reservoir metric learning. In *CVPR*, 2012.
  - [24] Z. Li, E. Gavves, T. Mensink, and C. G. Snoek. Attributes make sense on segmented objects. In *ECCV*, 2014.
  - [25] J. Lu, X. Zhou, Y.-P. Tan, Y. Shang, and J. Zhou. Neighborhood repulsed metric learning for kinship verification. *IEEE Trans. PAMI*, 36(2):331–345, 2014.
  - [26] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, 2012.
  - [27] A. Mignon and F. Jurie. PCCA: A new approach for distance learning from sparse pairwise constraints. In *CVPR*, 2012.
  - [28] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.
  - [29] M. Norouzi, D. M. Blei, and R. R. Salakhutdinov. Hamming distance metric learning. In *NIPS*, 2012.
  - [30] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR Workshops*, 2014.
  - [31] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover’s distance as a metric for image retrieval. *IJCV*, 40(2):99–121, 2000.
  - [32] R. Salakhutdinov and G. E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AISTATS*, 2007.
  - [33] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural computation*, 12(5):1207–1245, 2000.
  - [34] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *NIPS*, 2004.
  - [35] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng. Online and batch learning of pseudo-metrics. In *ICML*, 2004.
  - [36] L. Sharan, R. Rosenholtz, and E. Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009.
  - [37] G. Sharma and F. Jurie. Learning discriminative representation image classification. In *BMVC*, 2011.
  - [38] G. Sharma and F. Jurie. A novel approach for efficient SVM classification with histogram intersection kernel. In *BMVC*, 2013.
  - [39] G. Sharma, F. Jurie, and P. Perez. Learning non-linear SVM in input space for image classification. *HAL archives*, 2014.
  - [40] G. Sharma, F. Jurie, and C. Schmid. Expanded parts model for human attribute and action recognition in still images. In *CVPR*, 2013.
  - [41] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher vector faces in the wild. In *BMVC*, 2013.
  - [42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.
  - [43] L. Torresani and K.-C. Lee. Large margin component analysis. In *NIPS*, 2006.
  - [44] I. W. Tsang and J. T. Kwok. Distance metric learning with kernels. In *ICANN*, 2003.
  - [45] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2000.
  - [46] A. Vedaldi and K. Lenc. MatConvNet – Convolutional neural networks for MATLAB. In *ACM Multimedia*, 2015.
  - [47] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Trans. PAMI*, 34(3):480–492, 2012.
  - [48] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
  - [49] J. Wang, K. Markert, and M. Everingham. Learning models for object recognition from natural language descriptions. In *BMVC*, 2009.
  - [50] K. Q. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, 2006.
  - [51] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009.
  - [52] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In *NIPS*, 2002.
  - [53] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning Deep Features for Scene Recognition using Places Database. In *NIPS*, 2014.

# Appendix

## A. Normalization of the feature vectors

Tab. 3 gives the performance of the reference system which uses the full 4096 dimensional CNN feature, without any dimension reduction with any method. It compares different normalizations ( $\ell_1$  and  $\ell_2$ ) coupled with appropriate distances ( $\ell_1$ ,  $\ell_2$  and  $\chi^2$ ). Since the proposed method works with histograms we  $\ell_1$  normalize the features, to interpret them as histograms. While for other methods/baselines, we use  $\ell_2$  normalization. As Tab. 3 shows, different normalization coupled with appropriate different distance measure achieve essentially similar results.

Dataset	$\ell_2 + \ell_2$	$\ell_1 + \ell_1$	$\ell_1 + \chi^2$
Scene-15 [22]	85.0	84.5	84.6
Flickr Materials [36]	64.6	65.2	64.8
Leeds Butterflies [49]	96.0	96.0	96.0
Pascal VOC 2007 [10]	88.0	87.8	87.9
Human Attributes [37]	50.3	50.2	49.9
Oxford 102-Flowers [28]	73.2	73.6	73.9
Caltech UCSD Birds [48]	43.2	43.0	43.5

Table 3. Performance (mprec@ $K$  for  $K = 1$ ) for different normalization and distances on the CNN features for the reference system using full 4096 dimensional CNN feature.  $\ell_2 + \ell_2(\ell_1 + \ell_1)$  means the features were  $\ell_2(\ell_1)$  normalized and compared with  $\ell_2(\ell_2)$  distance while  $\ell_1 + \chi^2$  means they were  $\ell_1$  normalized and compared with  $\chi^2$  distance.

## B. Additional results for different parameter settings

Fig. 6–9 show the performance vs.  $K$  for different  $d \in \{8, 16, 32, 64\}$  and Fig. 10–12 show the performance of the method vs. the projection dimension  $d$  for different values of top retrieved examples  $K \in \{1, 10, 20, 30\}$ . The results support the discussion in the paper – the proposed method (NML) improves the performance over the baselines in most of the datasets for most of the settings. Otherwise, it is as good as the best baseline and only in very few cases, is worse than any baseline.

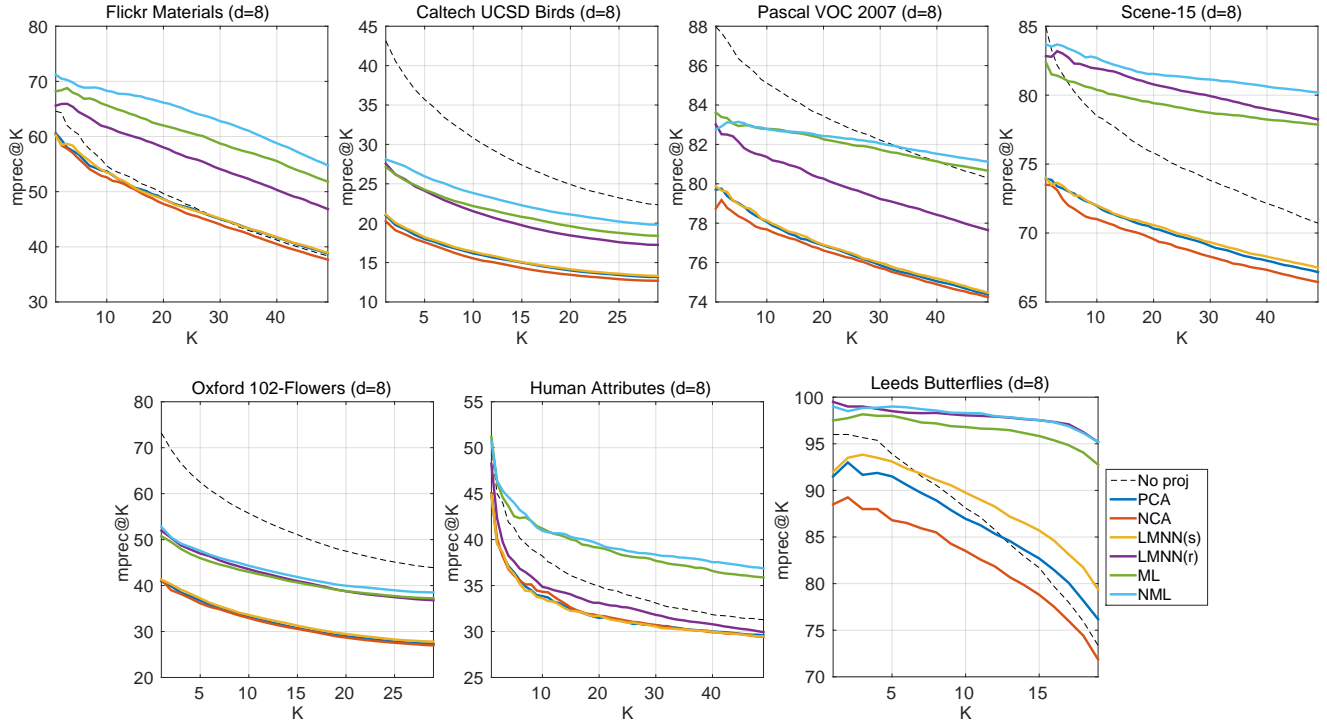


Figure 6. Comparisons of methods on the different datasets for  $K \in [1, \max(50, n^+ - 1)]$  and  $d = 8$ .

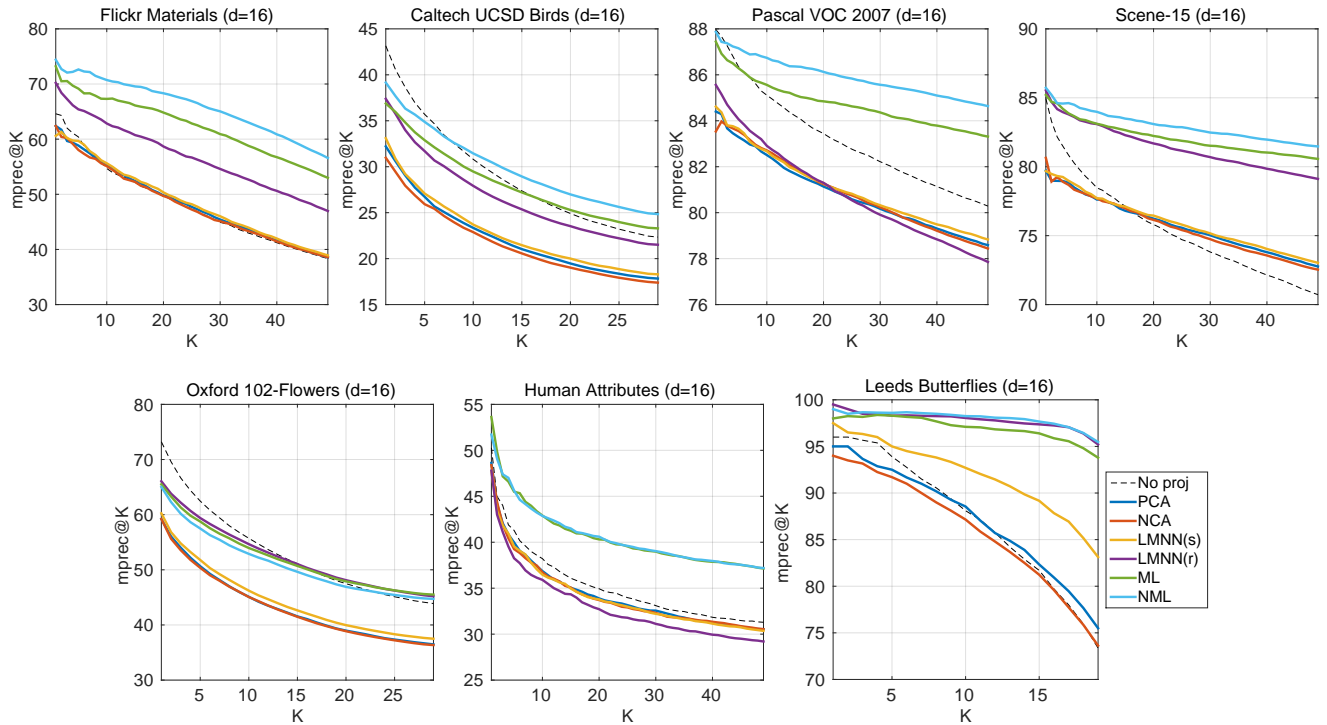


Figure 7. Comparisons of methods on the different datasets for  $K \in [1, \max(50, n^+ - 1)]$  and  $d = 16$ .

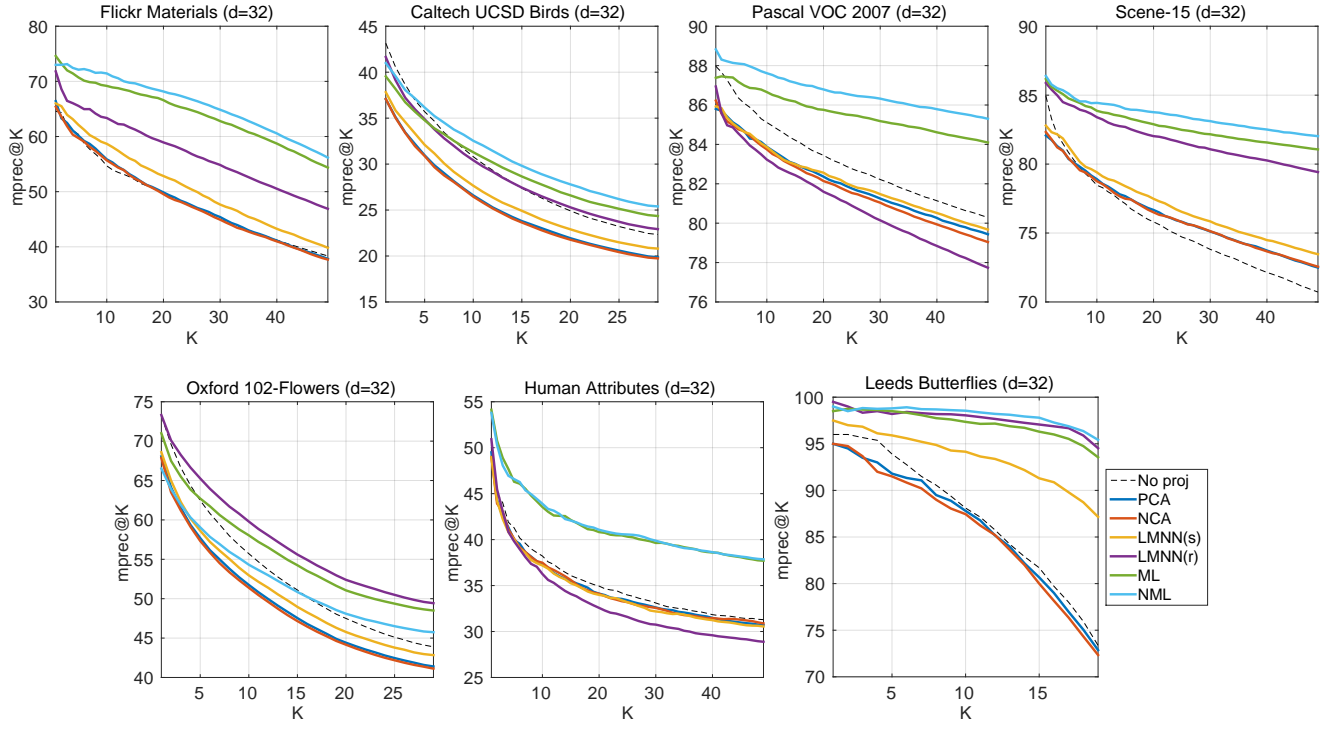


Figure 8. Comparisons of methods on the different datasets for  $K \in [1, \max(50, n^+ - 1)]$  and  $d = 32$ .

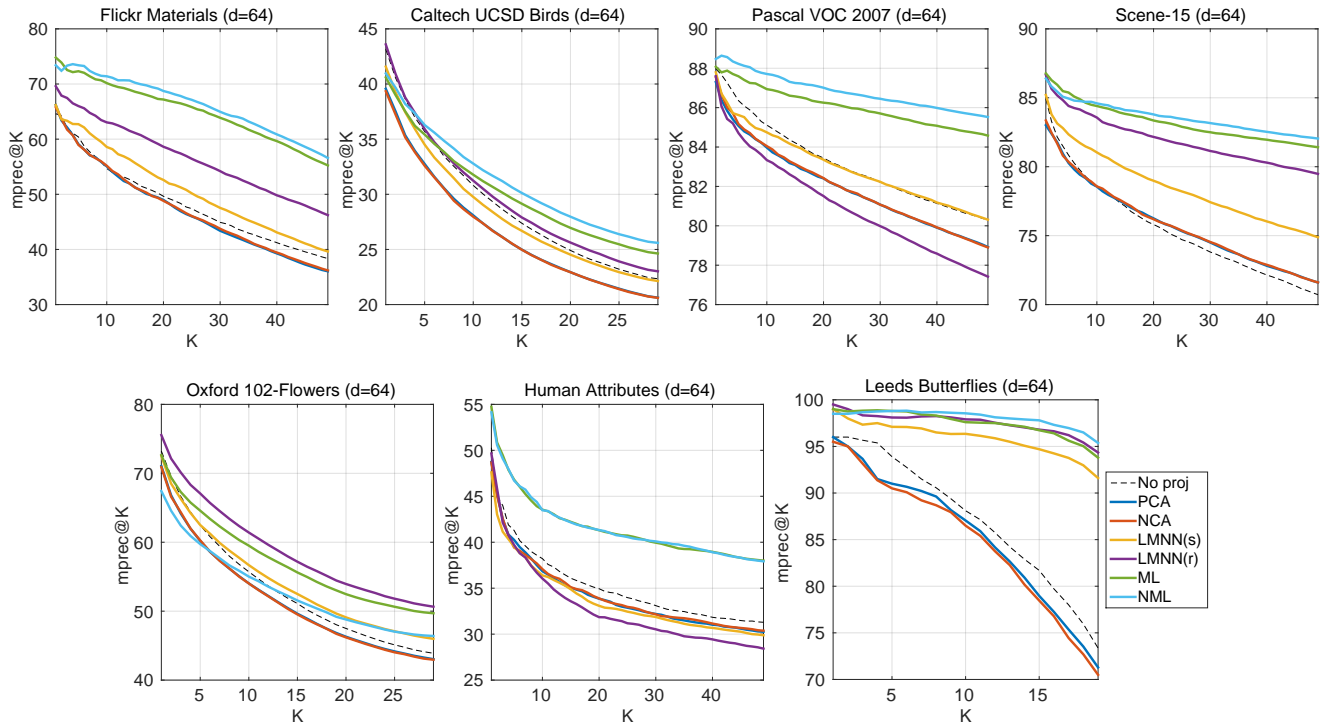


Figure 9. Comparisons of methods on the different datasets for  $K \in [1, \max(50, n^+ - 1)]$  and  $d = 64$ .



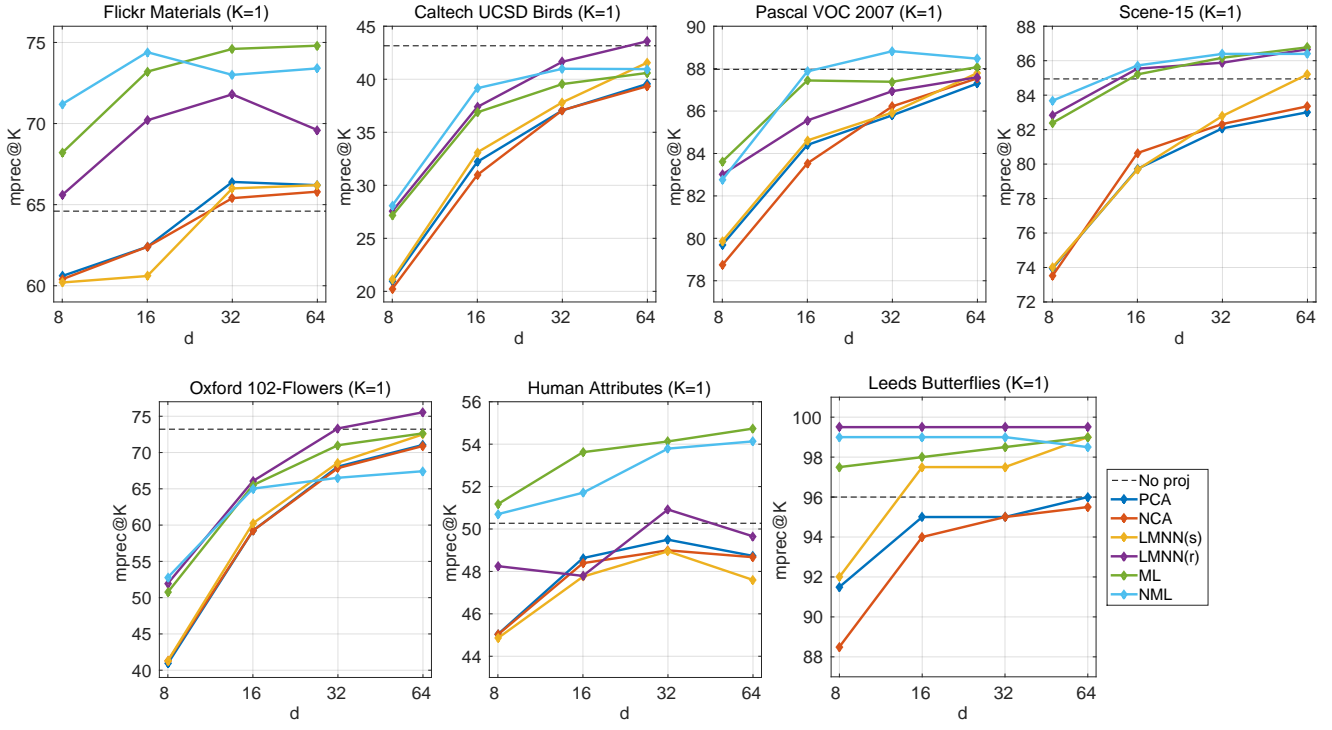


Figure 10. Comparisons of methods on the different datasets for  $K = 1$  and  $d \in \{8, 16, 32, 64\}$ .

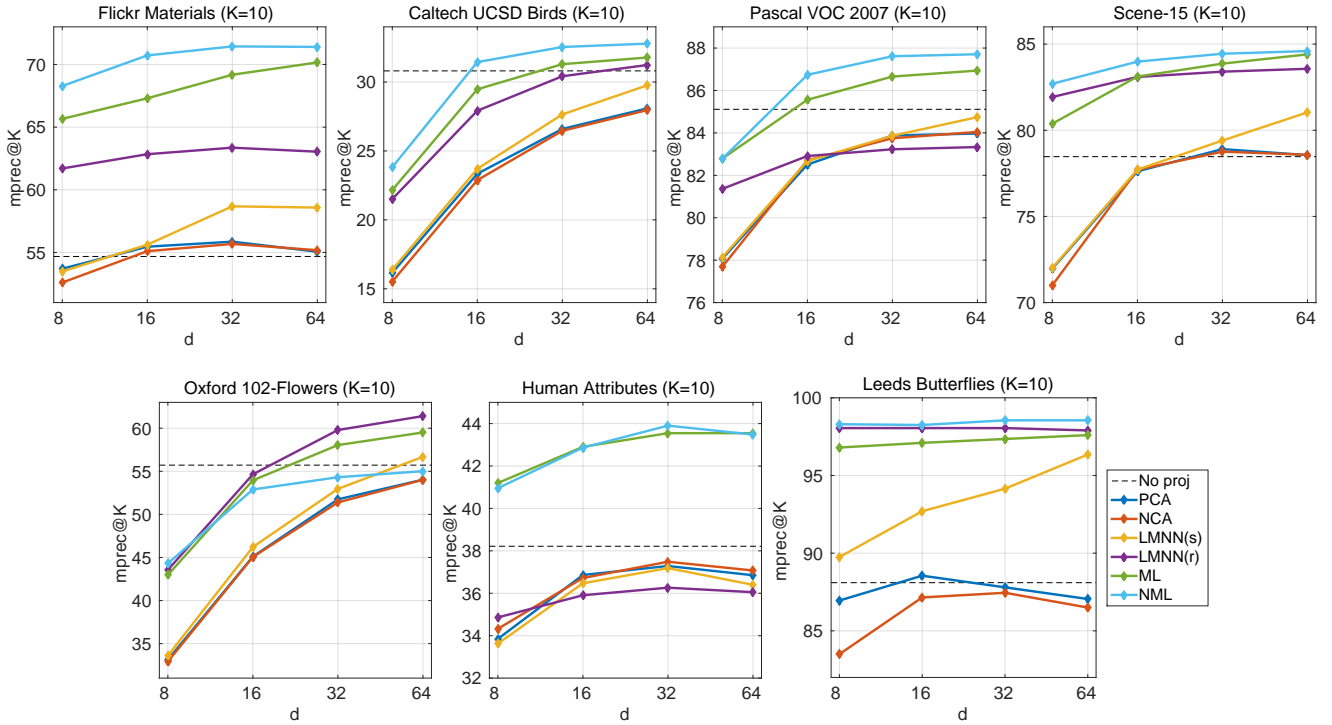


Figure 11. Comparisons of methods on the different datasets for  $K = 10$  and  $d \in \{8, 16, 32, 64\}$ .

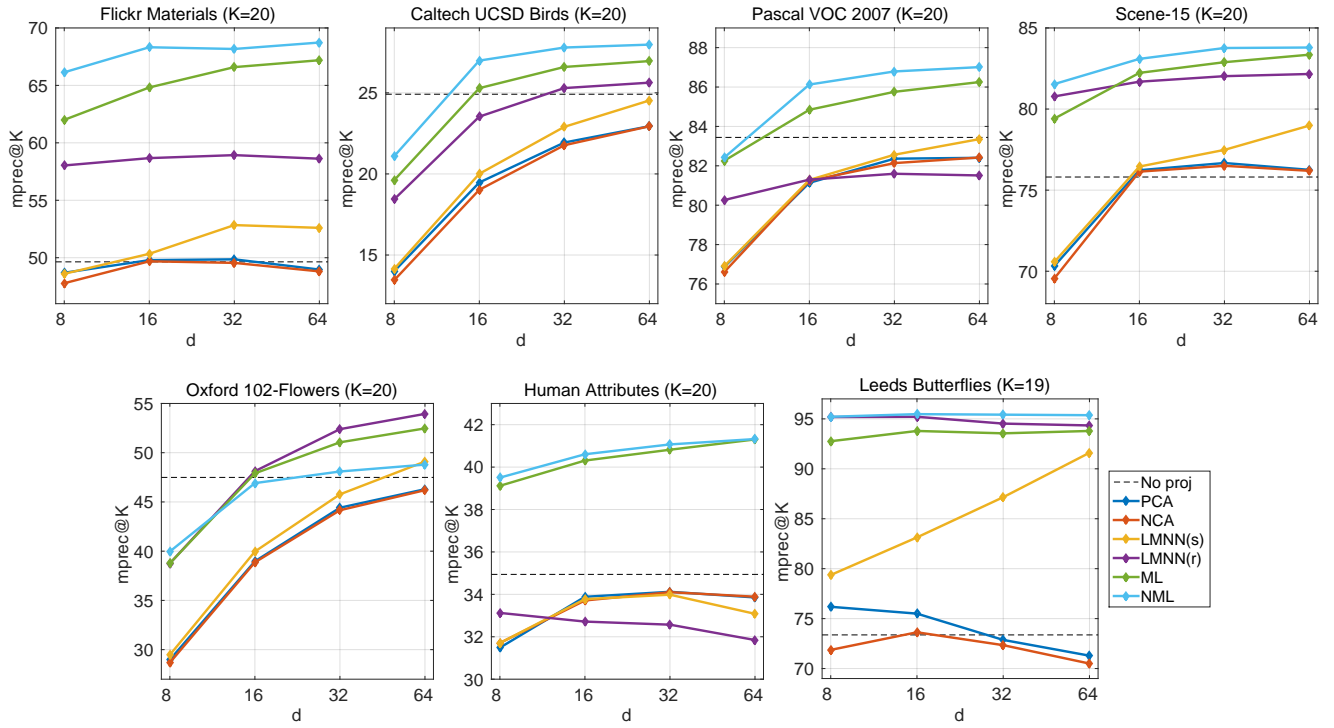


Figure 12. Comparisons of methods on the different datasets for  $K = 20$  (19, for datasets with only 20 positive images per class) and  $d \in \{8, 16, 32, 64\}$ .

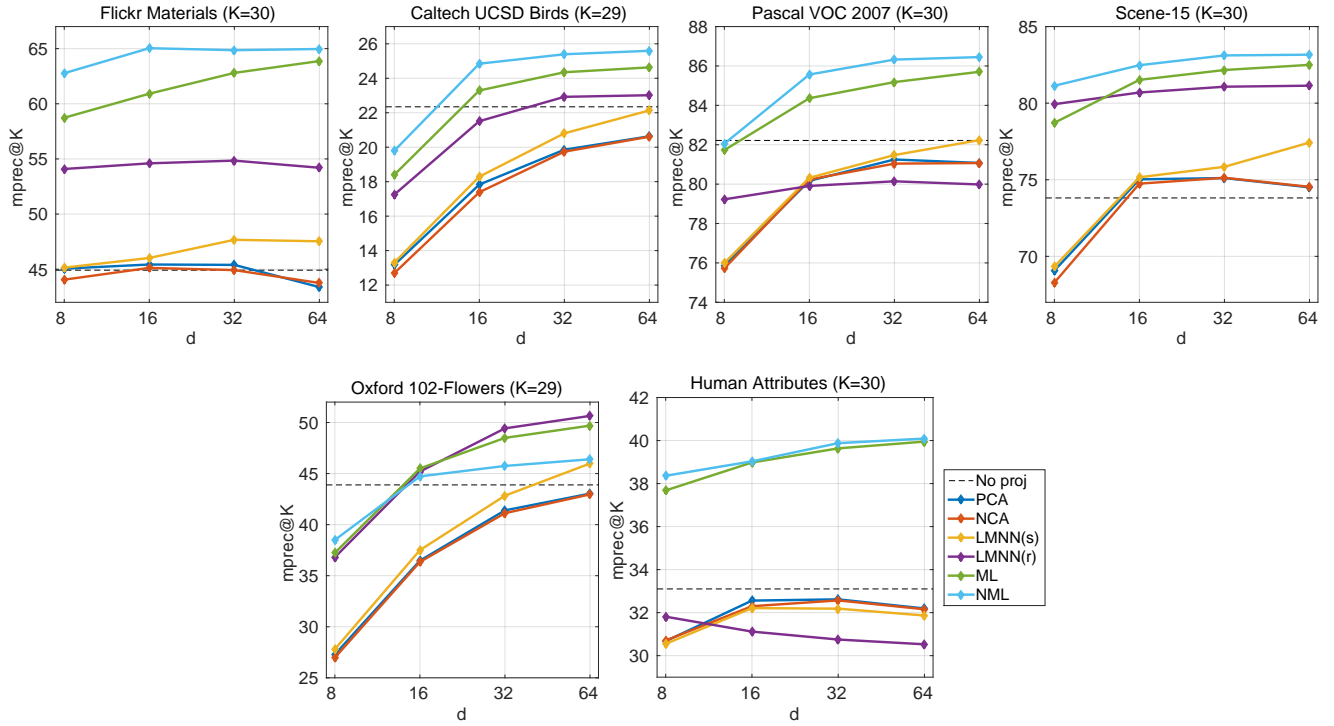


Figure 13. Comparisons of methods on the different datasets for  $K = 30$  (29, for datasets with only 30 positive images per class) and  $d \in \{8, 16, 32, 64\}$ .